Variable Coded Modulation (VCM) Cognitive Radio Higher-Order Constellation Receiver Operation

Andre Tkacenko[*], Michael J. Kilzer[†], Matthew D. Chase[‡], Susan C. Clancy[‡], Hua Xie[*], Jordan L. Torgerson[*], and Gregory J. Miles[*]

ABSTRACT. — In this article, we describe the receiver operation for a software-defined radio (SDR) configured to accommodate variable coded modulation (VCM) and adaptive coded modulation (ACM) formats. Specifically, we detail an SDR generated via GNU Radio used to receive VCM/ACM type signals. The focus here is on higher-order constellations used for VCM/ACM, consisting of M-ary amplitude and phase-shift keying (APSK) constellations, for $M = 16$, 32, and 64. We convey the receiver topology used to demodulate these higher-order constellations and show how this differs from a more conventional topology used to demodulate lower-order constellations, such as binary phase-shift keying (BPSK) and quadrature phase-shift keying (QPSK). Demodulation of the higher-order constellations considered here is shown for samples received through the following three methods: internally with GNU Radio, externally from a Universal Software Radio Peripheral (USRP) connected to a GNU Radio transmitter, and externally from the Jet Propulsion Laboratory (JPL) Space Telecommunications Radio System (STRS) field-programmable gate array (FPGA) hardware.

I. Introduction

Variable coded modulation (VCM), sometimes also referred to as adaptive coded modulation (ACM), is a digital communication format that allows for the change of coding and modulation in the course of a communication session in order to adapt the

[*]Communications Architectures and Research Section.

[†]Tracking Systems and Applications.

[‡]Flight Communications Systems.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

underlying information data rate to dynamic link conditions. In contrast to a conventional communication system that uses a fixed coding and modulation scheme designed to accommodate the worst-case link conditions, VCM/ACM can significantly increase the effective data throughput when the transmitter is appropriately adaptively configured to fully utilize available link capacity.

The VCM/ACM protocol we investigate here incorporates coding and modulation formats standardized by the Consultative Committee for Space Data Systems (CCSDS). In particular, convolutional, Reed-Solomon, concatenated, turbo, and low-density parity-check (LDPC) codes as described in [1] are used, in addition to serial concatenated convolutional codes (SCCC) as described in [2]. Modulation formats considered are M-ary phase-shift keying (PSK) constellations, as described in [1], for $M = 2$ (binary PSK or BPSK), $M = 4$ (quadrature PSK or QPSK), and $M = 8$ (8-PSK). In addition to these lower-order modulation formats, higher-order ones are considered, such as M-ary amplitude and phase-shift keying (APSK) constellations, as described in [2], for $M = 16$ (16-APSK), $M = 32$ (32-APSK), and $M = 64$ (64-APSK).

Physical layer (PL) frames are used to switch between different coding and modulation formats for VCM/ACM. The method used here is similar to that used for the Digital Video Broadcasting - Satellite - Second Generation (DVB-S2), standardized by the European Telecommunications Standards Institute (ETSI) and described in [3]. Each frame consists of a header field, which includes a synchronization word followed by a VCM/ACM code/modulation descriptor, along with a payload field, consisting of encoded data modulated with the appropriate constellation.

The VCM PL modes to be supported by the Space Telecommunications Radio System (STRS) at the Jet Propulsion Laboratory (JPL) are described in Table 1. Among the modes supported is one consisting of a pseudorandom binary sequence (PRBS) (Mode 0), which is used for testing. All LDPC codes here are accumulate-repeat-4-jagged-accumulate (AR4JA) type codes [4]. The (8160,7136) C2 code is an expurgated, shortened, and extended version of a basic (8176,7156) LDPC code, as described in [1].

A software-defined radio (SDR) receiver, compiled in GNU Radio[1], has been generated to demodulate and receive the VCM/ACM PL frames to be supported by the STRS. Unlike that used in [5], which is tailored for the lower-order constellation modes (i.e., modes with BPSK, QPSK, and 8-PSK), this cognitive radio receiver can accommodate both the higher-order constellation modes (i.e., modes with 16-APSK, 32-APSK, and 64-APSK) as well.

A. Outline

In Section II, we derive a signal model for the VCM/ACM physical layer (PL) and elaborate further on the intricacies of the PL frame. Challenges with demodulating higher-order constellations are discussed in Section II-A.

---

[1]GNU Radio is a free software development toolkit that provides signal processing blocks to implement software-defined radios and signal-processing systems.

Table 1. VCM PL modes to be supported by the STRS. (Input consists of bits coming into PL frame constructor, whereas output consists of symbols present in the PL frame payload.)

VCM Mode	Modulation	Code	Code Rate	Input Length (bits)	Output Length (symbols)
0	BPSK	PRBS	1	1024	1024
1	BPSK	Turbo	1/6	1784	10704
2	BPSK	Turbo	1/4	1784	7136
3	BPSK	Turbo	1/3	1784	5352
4	BPSK	LDPC	1/2	1024	2048
5	BPSK	LDPC	2/3	1024	1536
6	BPSK	LDPC	4/5	1024	1280
7	BPSK	C2	223/255	7136	8160
8	QPSK	LDPC	1/2	1024	1024
9	QPSK	LDPC	2/3	1024	768
10	QPSK	LDPC	4/5	1024	640
11	QPSK	C2	223/255	7136	4080
12	8-PSK	LDPC	1/2	1024	683[a]
13	8-PSK	LDPC	2/3	1024	512
14	8-PSK	LDPC	4/5	1024	427[a]
15	8-PSK	C2	223/255	7136	2720
16	16-APSK	LDPC	1/2	1024	512
17	16-APSK	LDPC	2/3	1024	384
18	16-APSK	LDPC	4/5	1024	320
19	16-APSK	C2	223/255	7136	2040
20	32-APSK	LDPC	1/2	1024	410[a]
21	32-APSK	LDPC	2/3	1024	308[a]
22	32-APSK	LDPC	4/5	1024	256
23	32-APSK	C2	223/255	7136	1632
24	64-APSK	LDPC	1/2	1024	342[a]
25	64-APSK	LDPC	2/3	1024	256
26	64-APSK	LDPC	4/5	1024	214[a]
27	64-APSK	C2	223/255	7136	1360

[a] Zero-padding is applied to the end of each VCM frame to produce an integer number of modulated symbols.

In Section III, we describe the receiver architecture used to accommodate the higher-order constellation modes. Differences with conventional architectures used for lower-order constellation modes are conveyed in Section III-A.

Examples of VCM/ACM higher-order constellation demodulation are given in Section IV. Specifically, examples of demodulation with samples received internally with GNU Radio are given in Section IV-A, whereas demodulation examples using samples received externally from a Universal Software Radio Peripheral (USRP) connected to a GNU Radio transmitter and from the STRS field-programmable gate array (FPGA) hardware are given in Sections IV-B and IV-C, respectively.

Finally, concluding remarks are made in Section V. There, we detail next steps to improve the capabilities of the GNU Radio based cognitive radio receiver.

II. VCM/ACM Physical Layer Signal Model

Currently, the VCM PL frames accommodated by the STRS consist of a single codeword. Typically, PL frames are contiguous, but need not be so in practice. The format for a VCM/ACM PL frame is as shown in Figure 1. Each VCM PL frame consists of a header (HDR) section, consisting of a frame marker (FM) and a frame descriptor (FD), followed by the payload (PLD) portion, which consists of a single codeword, encoded according to the mode shown in Table 1.

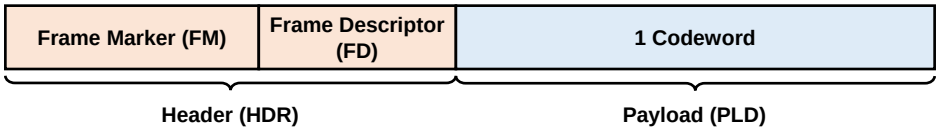

Figure 1. Structure of VCM/ACM PL frame.

The entire HDR section consists of BPSK symbols, whereas the PLD codeword is modulated according to the symbols for the particular VCM mode from Table 1. Here, the BPSK symbols are rotated by $\frac{\pi}{4}$ radians, compared with the more conventional BPSK symbols $\{-1, 1\}$, in order to make the BPSK constellation a subset of all other VCM mode constellations. Specifically, the BPSK constellation used here is as shown in Figure 2.

The FM is the same for every VCM PL frame and serves as a synchronization word for the receiver to ascertain the start of a frame. In hexadecimal, the FM consists of the following 256-bit field mapped to BPSK symbols:

$$\begin{array}{cccc} \text{FB44} & \text{1F1D} & \text{BDD7} & \text{76F2} \\ \text{3479} & \text{DA10} & \text{B4B3} & \text{AB9D} \\ \text{7D75} & \text{BF1E} & \text{70CA} & \text{EECF} \\ \text{9485} & \text{CDEC} & \text{E51A} & \text{F141} \end{array} \tag{1}$$

The FD is a 64-bit field mapped to BPSK symbols that uniquely describes the VCM mode corresponding to the PL frame. It serves the purpose of informing the receiver of the specific VCM mode for the PL frame. It consists of a 7-bit information field used to specify code, modulation, pilot on/off, and frame length. The FD is then encoded into a 64-bit vector. The elements of the 7×1 information bit vector

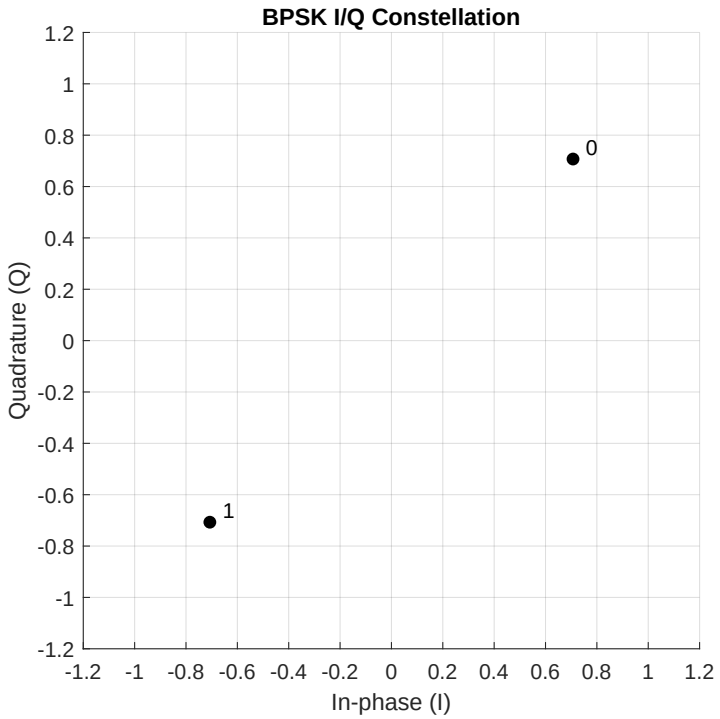

Figure 2. In-phase (I) / quadrature (Q) plot of the BPSK constellation used by the STRS. (The I/Q symbol map $\{0, 1\}$ is shown in the plot.)

$\mathbf{b} \triangleq \begin{bmatrix} b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 \end{bmatrix}^T$ are described in Table 2. Here, b_6 allows for the insertion of distributed pilot symbols after each codeword, while b_7 allows for the presence of multiple codewords in a PL frame. Currently, the STRS supports exclusively $b_6 = 0$ and $b_7 = 0$, and thus, each VCM PL frame payload consists of a single codeword with no pilot symbols, as shown in Figure 1.

Table 2. Characterization of the 7×1 information bit vector $\mathbf{b}$ from the frame descriptor (FD) of the VCM PL frame.

Bit Number	Content
b_1, b_2, b_3, b_4, b_5	VCM mode index (0 to 31), with b_1 being the most significant bit
b_6	Distributed pilot on (= 1) / off (= 0)
b_7	Frame-length option long (= 1) or short (= 0)

The 7×1 information bit vector $\mathbf{b}$ is mapped to the 64-bit FD as shown in Figure 3. The first 6 bits of $\mathbf{b}$ are mapped to a 32-bit vector $\mathbf{u}$ via the 32×6 linear mapping $\mathbf{G}$.

Here, $\mathbf{G}$ is given by the following:

$$\mathbf{G} \triangleq \begin{bmatrix} 0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1\,0\,1 \\ 0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1\,0\,0\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \\ 1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1\,1 \end{bmatrix}^T.$$

The elements of $\mathbf{u}$ are then input to a 32-to-1 multiplexer (MUX), each output of which is passed on to an exclusive-or (XOR) (i.e., modulo-2 addition) with information bit b_7, as well as a subsequent 2-to-1 MUX. The second input to the 2-to-1 MUX is the output from the XOR gate. The output from the 2-to-1 MUX is then sent to a 1-to-64 demultiplexer (DEMUX), which forms the 64×1 vector $\mathbf{v}$, whose components are as follows:

$$\mathbf{v} \triangleq \begin{bmatrix} u_1 & u_1 \oplus b_7 & u_2 & u_2 \oplus b_7 & \cdots & u_{32} & u_{32} \oplus b_7 \end{bmatrix}^T.$$

Here, $\oplus$ denotes modulo-2 addition. Finally, the vector $\mathbf{v}$ is scrambled by the 64×1 randomizer word $\mathbf{w}$, as shown in Figure 3, with the XOR operation at the right of the figure. The randomizer word $\mathbf{w}$ is given as follows:

$$\mathbf{w} \triangleq \begin{bmatrix} 0111000110011101100000111100100101010011010000100010110111111010 \end{bmatrix}^T.$$

The output of the system in Figure 3 is the 64×1 vector of encoded FD bits $\mathbf{x}$, which is given by $\mathbf{x} \triangleq \mathbf{v} \oplus \mathbf{w}$.

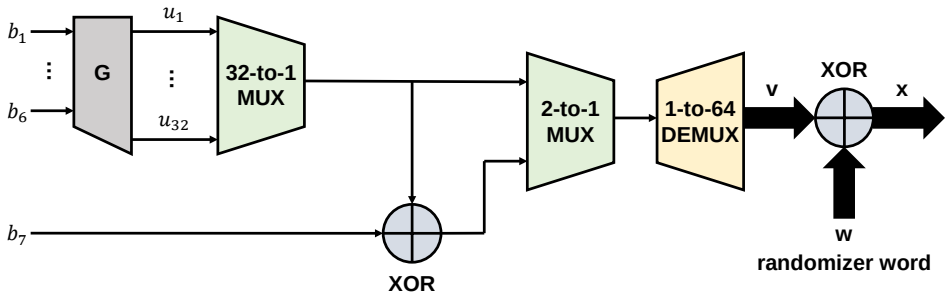

Figure 3. Generation of the 64-bit frame descriptor (FD) from the 7-bit information vector $\mathbf{b}$. (MUX stands for multiplexer, DEMUX stands for demultiplexer, and XOR stands for exclusive-or.)

The symbols appearing in the VCM PL frame shown in Figure 1 are shaped by a root-raised-cosine (RRC) filter $p(x)$, given by the following expression:

$$p(x) \triangleq \begin{cases} R_{\text{sym}}\left(1+\beta\left(\frac{4}{\pi}-1\right)\right), & x = 0, \\ \frac{R_{\text{sym}}\beta}{\sqrt{2}}\left[\left(1+\frac{2}{\pi}\right)\sin\left(\frac{\pi}{4\beta}\right)+\left(1-\frac{2}{\pi}\right)\cos\left(\frac{\pi}{4\beta}\right)\right], & x = \pm\frac{1}{4\beta}, \\ R_{\text{sym}}\dfrac{\sin\left[\pi x\left(1-\beta\right)\right]+4\beta x\cos\left[\pi x\left(1+\beta\right)\right]}{\pi x\left[1-\left(4\beta x\right)^2\right]}, & \text{otherwise}, \end{cases} \tag{2}$$

where R_{sym} is the *symbol rate* and β is the *roll-off factor*, which can be in the interval $[0, 1]$. The pulse-shaped symbols are then upconverted by a carrier and transmitted. The receiver then downconverts the received waveform, resulting in a continuous-time, complex baseband signal $r(t)$. Nominally, $r(t)$ is as follows:

$$r(t) \triangleq Ae^{j(2\pi F_0 t+\theta_0)} \sum_{k\in\mathcal{I}} d_k p(R_{\text{sym}} t - (k+\epsilon)) + w(t)\,. \tag{3}$$

Here, the quantities appearing in Equation (3) are the following:

- A: amplitude factor,
- F_0: carrier frequency offset (CFO) or residual Doppler,
- θ_0: carrier phase offset (CPO), satisfying $\theta_0 \in [0, 2\pi)$,
- $\mathcal{I}$: index set of symbols over VCM PL frames,
- d_k: k-th symbol (can correspond to either a HDR symbol or a PLD symbol),
- ϵ: symbol timing offset (STO), satisfying $\epsilon \in [0, 1)$,
- $w(t)$: continuous-time circular complex additive white Gaussian noise (AWGN) process with zero mean and power spectral density N_0.

The signal $r(t)$ is then low-pass filtered down to a bandwidth of F_s and then sampled at a rate of F_s (i.e., sampled at $t = n/F_s$). Assuming the low-pass filter is approximately ideal, we obtain the discrete-time complex baseband sequence $r[n]$ given as follows:

$$r[n] \triangleq Ae^{j(2\pi f_0 n+\theta_0)} \sum_{k\in\mathcal{I}} d_k p\Big(\frac{n}{K} - (k+\epsilon)\Big) + w[n]\,. \tag{4}$$

In Equation (4), the quantities appearing are the following:

- $f_0 \triangleq \frac{F_0}{F_s}$: normalized CFO (NCFO) or normalized residual Doppler,
- $K \triangleq \frac{F_s}{R_{\text{sym}}}$: oversampling factor (i.e., the number of samples per symbol),
- $w[n]$: discrete-time circular complex AWGN process with zero mean and variance $\sigma_w^2 \triangleq N_0 F_s$.

A. VCM/ACM Higher-Order Constellations and Challenges with Demodulation

From the receiver discrete-time sequence $r[n]$ given in Equation (4), we would like to recover the set of symbols $\{d_k\}_{k\in\mathcal{I}}$. This requires both carrier and symbol synchronization, which must occur in the presence of noise. Specifically, carrier synchronization involves estimating the NCFO f_0 and CPO θ_0 and removing their effects on the received sequence $r[n]$ shown in Equation (4). In addition, symbol synchronization involves estimating and undoing the effects of the oversampling factor

K from Equation (4), which may slightly deviate from a nominal value, say K_{nom}, which is usually an integer value such as 4.

Typically, carrier and symbol synchronization are carried out separately [6]. Carrier synchronization is usually carried out using a Costas loop, while symbol synchronization is carried out via a Gardner loop or a data transition tracking loop (DTTL) [6]. Furthermore, typically carrier synchronization is done prior to symbol synchronization. However, a more recent common practice for GNU Radio based receivers is to reverse this order. In addition, symbol synchronization is carried out using a polyphase filter bank based approach, as described in [7], which combines symbol synchronization with matched filtering.

The polyphase filter bank based symbol synchronizer can accommodate a wide variety of modulations, including non-constant envelope modulation formats such as APSK and quadarature amplitude modulation (QAM), and can successfully achieve lock even in noisy environments. However, a Costas loop based carrier synchronizer used after symbol synchronization is specifically tailored for constant envelope modulations, such as PSK. While a Costas loop can track non-constant envelope modulations, such as 16-QAM, this can only typically be done successfully when the number of carrier lock points is small [6] (such as 4 in the case of 16-QAM). For APSK modulations, which have different lock points depending on the constellation ring under consideration, carrier synchronization using a conventional Costas loop implemented as is cannot be achieved.

To elucidate the issues pertaining to carrier synchronization for APSK modulations, the constellations for 16/32/64-APSK used by the STRS are shown in Figure 4. As can be seen, the APSK constellations consist of the following rings and number of points in each ring:

- 16-APSK: 4 points in ring with radius $1/2.75 = 0.\overline{36}$, 12 points in ring with radius 1,
- 32-APSK: 4 points in ring with radius $1/4.79 \approx 0.2088$, 12 points in ring with radius $2.72/4.79 \approx 0.5678$, 16 points in ring with radius 1,
- 64-APSK: 4 points in ring with radius $1/6.31 \approx 0.1585$, 12 points in ring with radius $2.73/6.31 \approx 0.4326$, 20 points in ring with radius $4.52/6.31 \approx 0.7163$, 28 points in ring with radius 1.

As mentioned earlier, Costas loops used for carrier synchronization are designed to only accommodate PSK modulations [6]. As such, a Costas loop used on any of the APSK modulations shown in Figure 4 may try to lock on to any of the PSK modulations corresponding to any of the constellation rings, each of which is a PSK constellation in itself. For example, for 32-APSK, a Costas loop may try to lock on to the 4 points in the inner ring, the 12 points in the middle ring, or the 16 points in the outer ring. The result will be an overall failure to achieve any reasonable carrier synchronization lock.

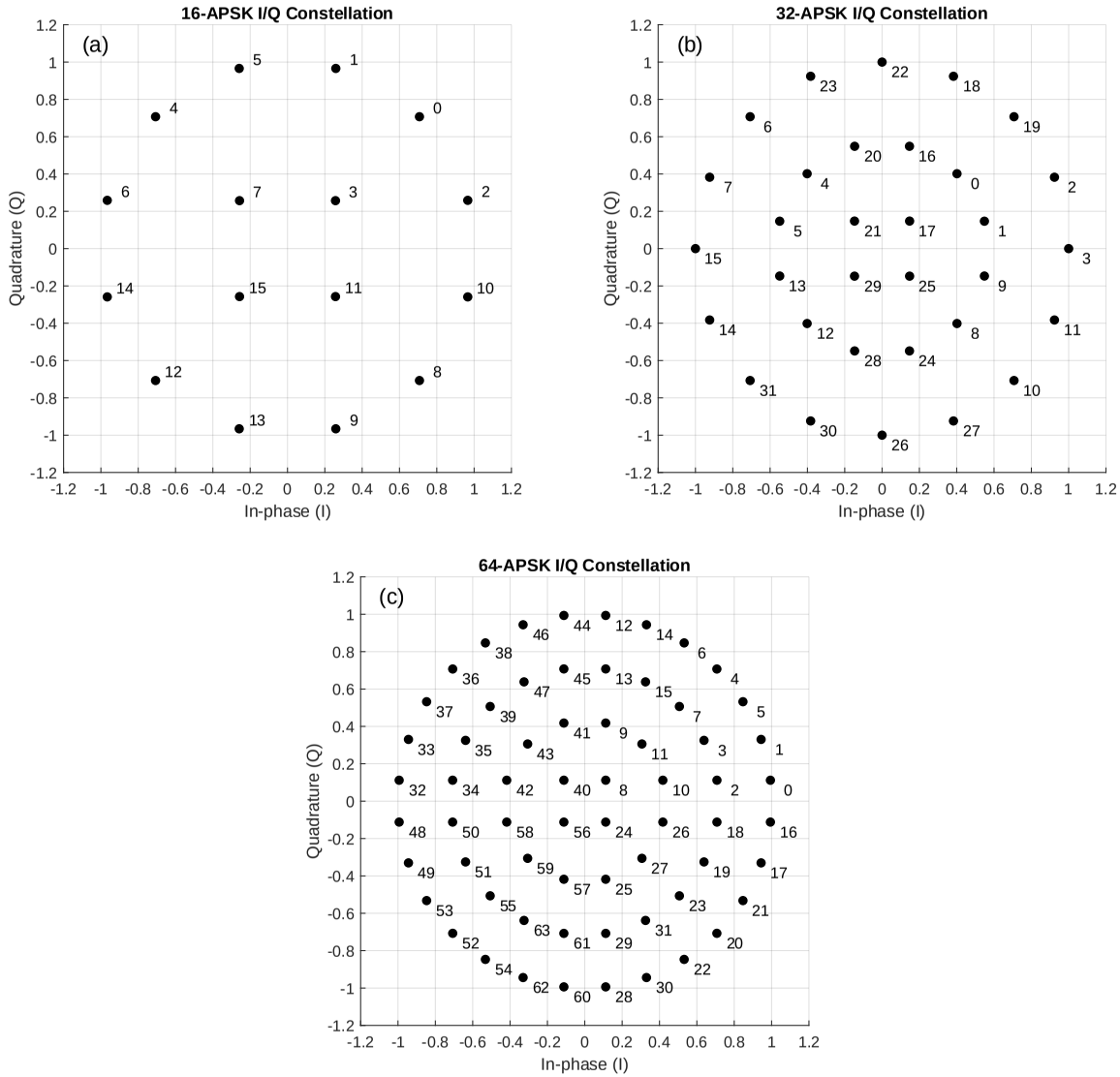

Figure 4. In-phase (I) / quadrature (Q) plot of the APSK constellations used by the STRS: (a) 16-APSK, (b) 32-APSK, and (c) 64-APSK. (The I/Q symbol map $\{0, 1, \ldots, N_c - 1\}$ is shown in each plot, where N_c denotes the number of constellation points.)

To overcome these issues, we exploit the fact that the FM bits appearing in the VCM PL frames are known at the receiver. In addition, it is assumed that the receiver will have knowledge of the type of RRC pulse shape $p(x)$ from Equation (2) used to transmit the data, specifically, the symbol rate R_{sym} and roll-off factor β. Finally, we assume that the receiver will have knowledge of the nominal oversampling factor K_{nom}, and that this will not deviate substantially from the true oversampling factor K observed in the received sequence $r[n]$ from Equation (4).

With this knowledge, the receiver can generate samples of the waveform it would expect to see from the FM portion of each VCM PL frame. This sequence of samples $s_{\text{FM}}[n]$ is given by the following:

$$s_{\text{FM}}[n] \triangleq \sum_{k \in \mathcal{I}_{\text{FM}}} d_k p\left(\frac{n}{K_{\text{nom}}} - k\right), \tag{5}$$

where $\mathcal{I}_{\text{FM}}$ denotes an index set corresponding to FM symbols in a given VCM PL frame. Without loss of generality, we could take $\mathcal{I}_{\text{FM}} = \{0, 1, \ldots, N_{\text{FM}} - 1\}$, where N_{FM} denotes the number of FM symbols in a VCM PL frame (here $N_{\text{FM}} = 256$), and take d_k to be the k-th FM symbol.

With $s_{\text{FM}}[n]$ formed as in Equation (5), the receiver can then correlate its received sequence $r[n]$ from Equation (4) with $s_{\text{FM}}[n]$. Local peaks in the magnitude of the correlation should then indicate the start of a VCM PL frame. The sample position and phase of the correlation at these local peaks can then be used to warm start synchronization loops by providing essentially pilot-based initial condition information for these loops [6]. Specifically, the sample position and phase of the correlation corresponding to each local peak can be used to provide an acquisition-type initial condition for the symbol and carrier synchronization loops used, respectively.

The sample position initial condition can be passed on to a polyphase filter-bank-based symbol synchronizer, while the correlation phase initial condition can be passed on to a Costas loop configured for the BPSK constellation in Figure 2. In order to maintain lock for the remainder of the VCM PL frame, we need only use a small loop bandwidth for both the polyphase filter-bank-based symbol synchronizer and the Costas loop carrier synchronizer. By using small loop bandwidth values, we can ensure that any loss of carrier lock due to APSK constellation symbols, which may be present in the VCM PL frame payload, will be minimized.

III. GNU Radio Based Cognitive Radio Receiver Architecture

A GNU Radio hierarchical block named `VCM_PHY_CMA_demod.grc` was created to extract soft symbol estimates from samples of the VCM PL frames. Referring to Equation (4), this block essentially takes in a sequence of received baseband samples $r[n]$ and outputs a sequence of estimates of the symbols d_k, which we denote here by $\widehat{d_k}$. For diagnostic purposes, the block also outputs the cross-correlation $\rho[n]$ between

the received sequence $r[n]$ and the expected FM sequence $s_{\mathrm{FM}}[n]$, given by

$$\rho[n] \triangleq r[n] \circledast s^*_{\mathrm{FM}}[-n] = \sum_{m=-\infty}^{\infty} r[m]\, s^*_{\mathrm{FM}}[m-n]\,,$$

where $\circledast$ is the glyph denoting the *discrete convolution* between two sequences [7].

A screenshot from the GNU Radio flowgraph of the hierarchical block `VCM_PHY_CMA_demod.grc` is shown in Figure 5. As can be seen, starting from the pad source input on the left, the received sequence $r[n]$ is first correlated with the expected FM sequence $s_{\mathrm{FM}}[n]$ using the **Correlation Estimator** block. The top output from this block passes the input sequence $r[n]$ as is, along with tags corresponding to the correlation peak position/phase to the symbol/carrier tracking loops, respectively, to aid in synchronization. The bottom output from the **Correlation Estimator** block passes along the cross-correlation values, which can be observed for diagnostic purposes, if so desired.

After passing through the **Correlation Estimator** block, the received sequence $r[n]$ is fed to a **Polyphase Clock Sync** block, which carries out symbol synchronization. Before being applied to a Costas loop for carrier synchronization, though, the output of the **Polyphase Clock Sync** block is fed to a constant modulus algorithm (CMA) [8] equalization block **CMA Equalizer**. The purpose of this blind equalization block [7] is to account for any multipath artifacts [6] that may be present in the received sequence $r[n]$, which may not be properly accounted for using the matched filter present in the **Polyphase Clock Sync** block. Specifically, if the pulse shaping filter $p(x)$ appearing in Equation (4) is replaced with an alternative $\widetilde{p}(x)$, representing the effects of a linear multipath channel [8], but the receiver assumes the matched filter to be configured for $p(x)$, then the CMA equalizer can accommodate this discrepancy. Use of the CMA equalizer improved the demodulation performance for samples obtained from the STRS FPGA hardware, which showed signs of in-band aliasing manifesting effectively as a multipath channel.

Carrier synchronization is then carried out from the output of the **CMA Equalizer** block. This is executed through the use of a Costas loop [7] configured for the BPSK constellation in Figure 2, which is used for the VCM PL HDR symbols. As the GNU Radio Costas loop block **Costas Loop** configured for BPSK applies for the conventional BPSK constellation $\{-1, 1\}$ [6] and not the $\pi/4$-rotated version shown in Figure 2 used for VCM/ACM here, **Multiply Const** blocks are used before and after the **Costas Loop** to perform pre/post-rotation, respectively, account for this phenomenon. After carrier synchronization is complete, the estimated soft symbols from the VCM PL frames are fed to the pad sink output on the right of Figure 5.

A. Differences with Conventional Architectures for Lower-Order Constellations

As mentioned earlier, the **Correlation Estimator** block from Figure 5 generates tags, corresponding to the correlation peak position/phase, which are then fed to the symbol/carrier tracking loops, respectively, to aid in synchronization. Specifically, the

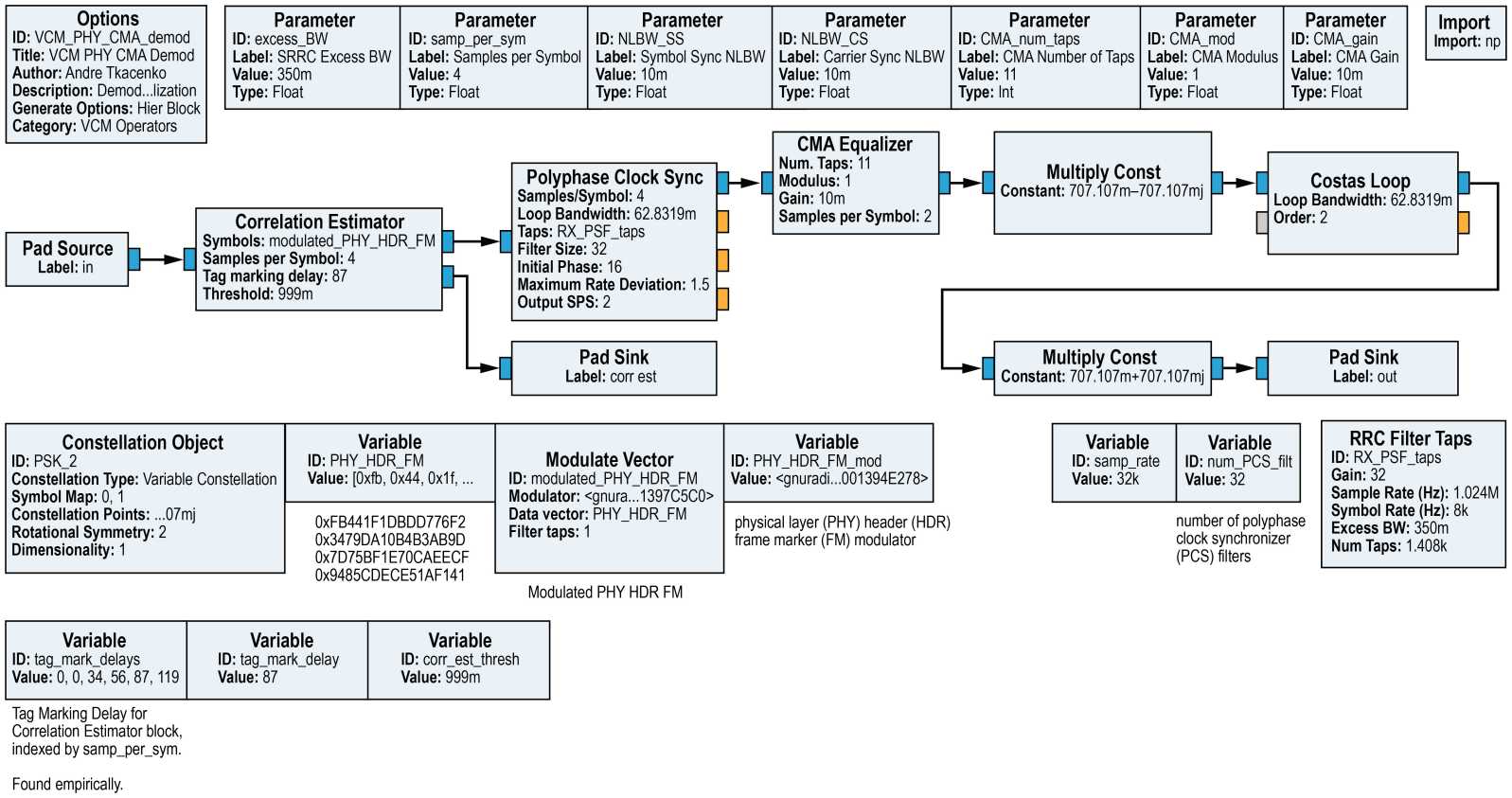

Figure 5. GNU Radio flowgraph screenshot of hierarchical block `VCM_PHY_CMA_demod.grc` used to generate soft symbol estimates $\widehat{d}_k$ from samples of the VCM PL frames $r[n]$. Individual blocks from this flowgraph, such as Pad Source, are written in bold font in this article, for sake of clarity.

correlation peak position tag is passed to the **Polyphase Clock Sync** block, while the correlation peak phase tag is applied to the **Costas Loop** block. These tags greatly reduce the amount of time required for symbol/carrier acquisition, and allow for the higher-order APSK constellations to be properly demodulated.

A more conventional demodulator block for the lower-order constellations (i.e., 8-PSK, QPSK, and BPSK) would omit the **Correlation Estimator** block and would use a **Costas Loop** block configured for the constellation used for the VCM PLD symbols. In this manner, the carrier/symbol tracking loops would be left to acquire using only the data presented to them. This was done, for example, in [5], to demodulate data from an International Space Station (ISS) pass. Specifically, for this pass (which consisted of BPSK, QPSK, and 8-PSK PLD data) a **Costas Loop** block configured for 8-PSK was successfully used to demodulate nearly all of the data. It should be noted also that for this pass, a **CMA Equalizer** block was not used to mitigate against any possible multipath effects.

IV. VCM/ACM Higher-Order Constellation Demodulation Examples

In this section, screenshots from GNU Radio of VCM/ACM higher-order constellation demodulation examples are shown to highlight the capabilities of the cognitive radio receiver. For each screenshot, the left-hand side shows the power spectral density (PSD) [6] of the received sequence (effectively $r[n]$ from Equation (4)), while the right-hand side shows an in-phase (I) / quadrature (Q) constellation plot of the estimated soft symbols (essentially the sequence $\widehat{d}_k$ described in Section III).

A. Samples Received Internally with GNU Radio

VCM formatted data was generated internally with GNU Radio and subsequently demodulated using the `VCM_PHY_CMA_demod.grc` hierarchical block described in Figure 5. Here, VCM Modes 17, 21, and 25 were considered as these employ higher-order constellations for their PLD data. Specifically, these modes use the following constellations for the PLD:

- VCM Mode 17: 16-APSK,
- VCM Mode 21: 32-APSK,
- VCM Mode 25: 64-APSK.

In order to emulate a somewhat realistic operational environment, channel degradations such as those shown in Equation (4) were used in formulating the received sequence of samples $r[n]$. These were generated in GNU Radio using a **Channel Model** block. For all of the results provided in this subsection, the following parameter values for this block were used: (parameter values are written in *italics* in this article, for sake of clarity)

- *Noise Voltage*: 0.15,
- *Frequency Offset*: 0.005,
- *Epsilon*: 1.0003.

Referring to Equation (4), the *Noise Voltage* parameter is proportional to the standard deviation σ_w of the AWGN process $w[n]$, the *Frequency Offset* parameter is equal to the NCFO f_0, and the *Epsilon* parameter is ratio between the transmitter and receiver sample clock frequencies, which is given by K_{nom}/K here.

In Figures 6, 7, and 8, GNU Radio screenshots from demodulating VCM Modes 17, 21, and 25 are shown, respectively. For all cases shown, a CMA equalizer with 11 taps was used, although the CMA equalizer was not technically needed in these cases, as there were no multipath channel effects. As can be seen, for all higher-order modes considered here, there were no issues with demodulation, and all soft symbols appear as they should. In each of the figures, from the I/Q constellation plots, it can be seen that there is a clustering around the BPSK constellation points from Figure 2. This clustering corresponds to the BPSK symbols appearing in the VCM PL frame HDR sections and is expected.

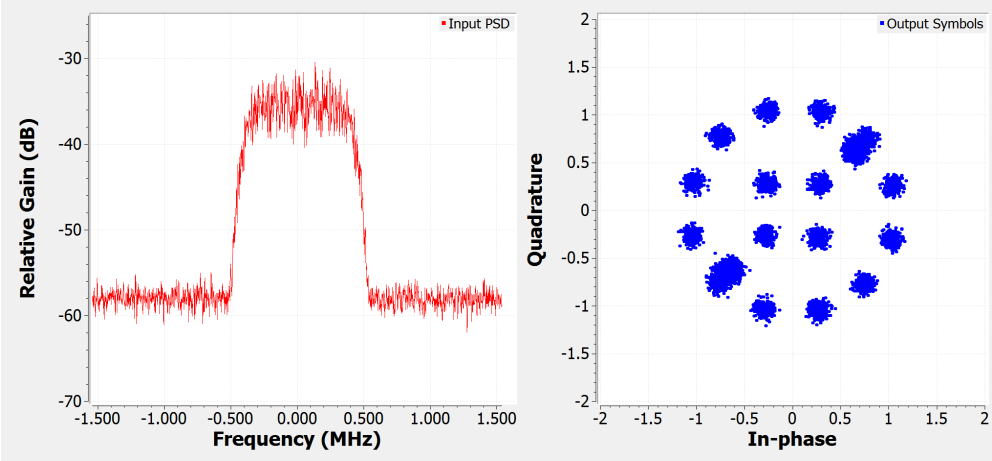

Figure 6. GNU Radio screenshot of VCM Mode 17 demodulation example using internally generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

B. Samples Received Externally from USRP Connected to GNU Radio Transmitter

As a gateway between testing the demodulation capabilities of the GNU Radio based VCM PL frame receiver in software, with internally generated data, and in hardware, using data captures from the STRS, demodulation was tested using samples received from a USRP. Specifically, a GNU Radio based transmitter was hooked up to a USRP, which transmitted VCM PL frames to another USRP, which was connected to the GNU Radio based receiver employing the `VCM_PHY_CMA_demod.grc` hierarchical block from Figure 5.

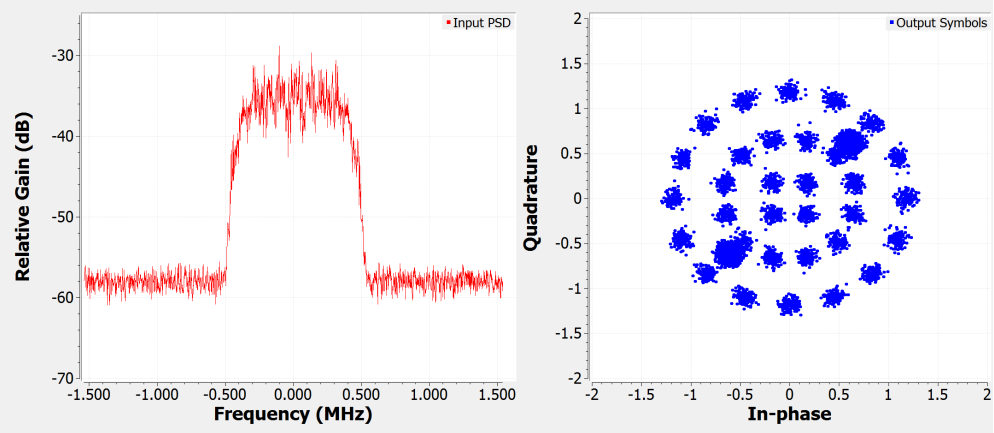

Figure 7. GNU Radio screenshot of VCM Mode 21 demodulation example using internally generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

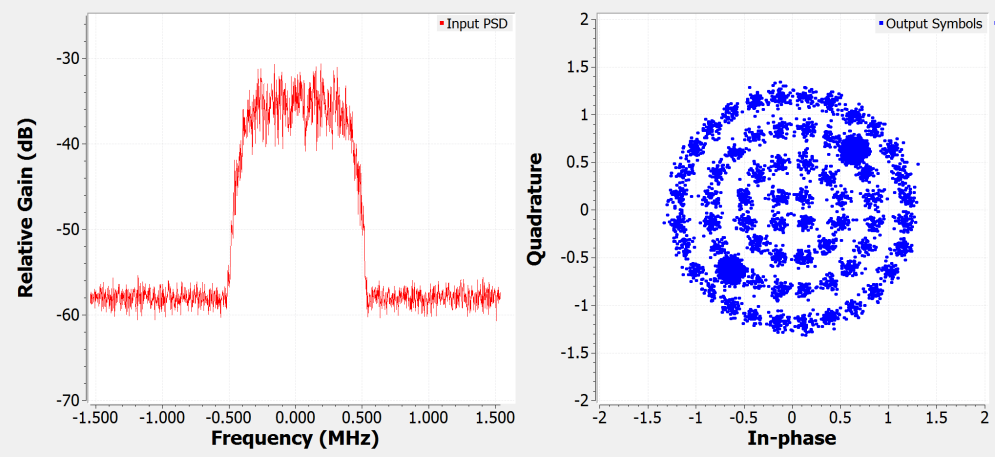

Figure 8. GNU Radio screenshot of VCM Mode 25 demodulation example using internally generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

As was done in Section IV-A, VCM Modes 17, 21, and 25 were considered here as these employ higher-order constellations for their PLD data. Furthermore, to emulate a realistic operational environment, channel degradations such as those from Equation (4) were generated at the GNU Radio transmitter via a **Channel Model** block prior to transmission using the first USRP. For all of the results provided in this subsection, the following parameter values for this block were used:

- Noise Voltage: 0.07,
- Frequency Offset: 0.005,
- Epsilon: 1.0003.

It should be noted that the *Noise Voltage* parameter was lowered to 0.07 here from the value 0.15 used in the previous subsection as the USRP transmit/receive chain imparts additional noise onto the received samples. With this lowered value for the

Noise Voltage, effectively, the same noise level was observed in this subsection as in the previous subsection, as evidenced by the PSD plots of the received sequences.

In Figures 9, 10, and 11, GNU Radio screenshots from demodulating VCM Modes 17, 21, and 25 are shown, respectively. For all cases considered, a CMA equalizer with 173 taps was used, although the CMA equalizer was not technically needed in these cases, as there were no multipath artifacts. As can be seen, for all higher-order modes considered here, the soft symbols observed were more noisy than those obtained in Section IV-A, however overall, there were no major issues with demodulation. From the I/Q constellation plots of each of the figures, it can be seen that there is a clustering around the BPSK constellation points from Figure 2. As mentioned earlier, this clustering corresponds to the BPSK symbols appearing in the VCM PL frame HDR sections and is expected.

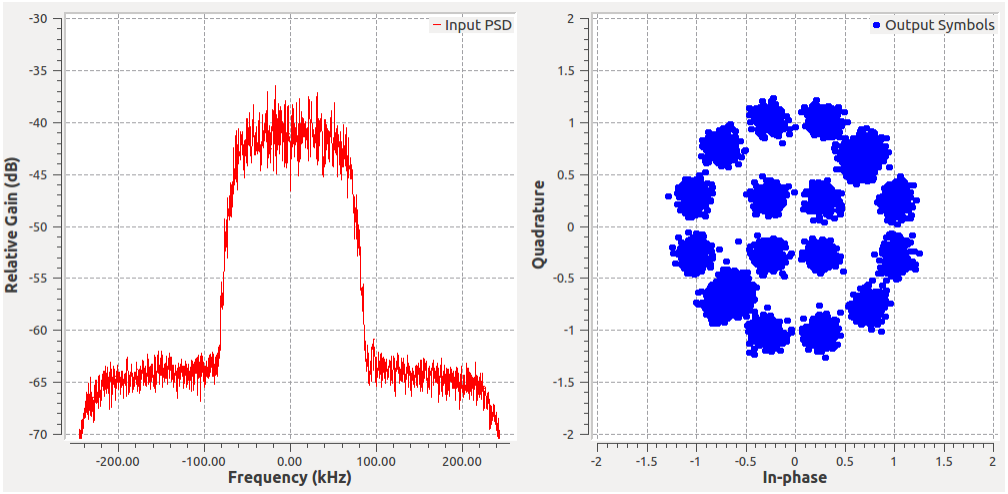

Figure 9. GNU Radio screenshot of VCM Mode 17 demodulation example using USRP generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

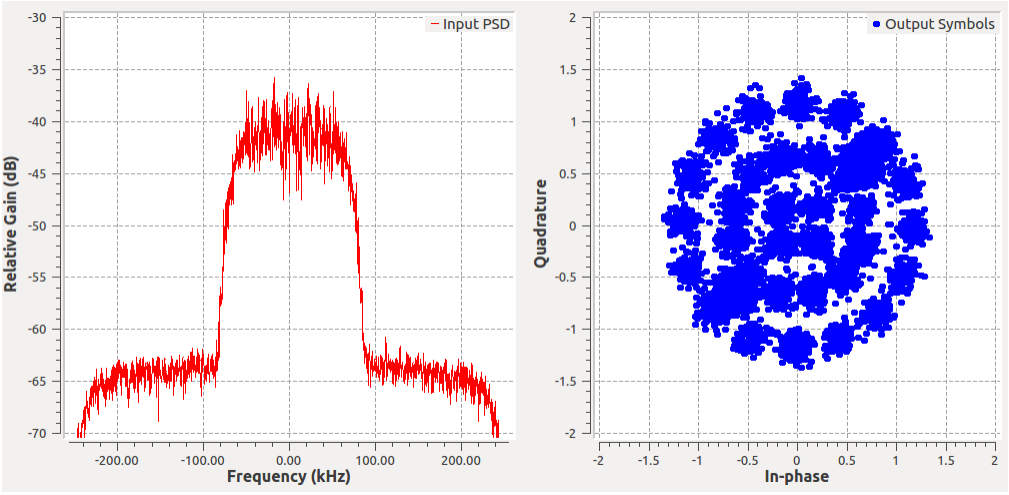

Figure 10. GNU Radio screenshot of VCM Mode 21 demodulation example using USRP generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

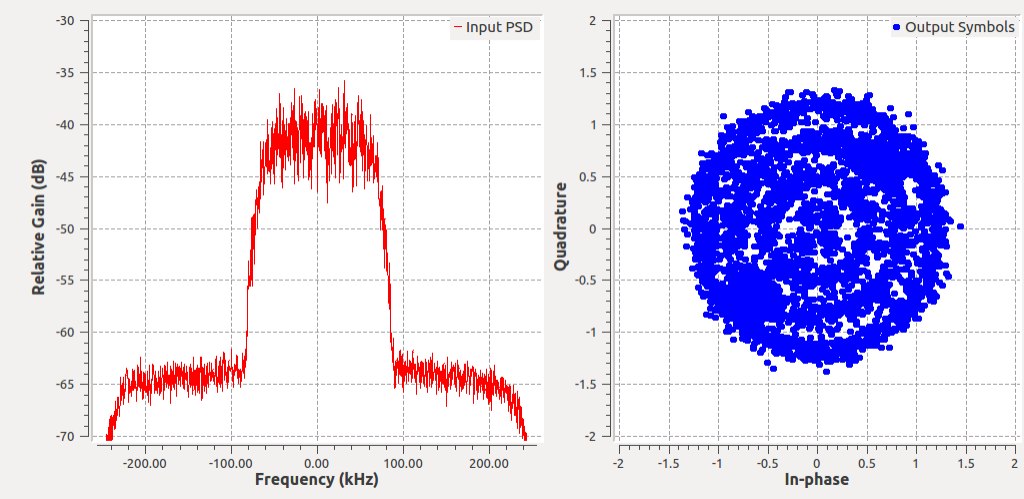

Figure 11. GNU Radio screenshot of VCM Mode 25 demodulation example using USRP generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

C. Samples Received Externally from STRS FPGA Hardware

Data captures obtained from the STRS were processed through the GNU Radio based cognitive receiver. While a wide variety of data was collected (from VCM Modes 0, 6, 7, 10, 11, 14, 15, 18, 19, 22, and 26), only the results from the higher-order constellation LDPC modes (18, 22, and 26) are shown here for brevity.

To highlight the impact of the CMA equalizer used in the `VCM_PHY_CMA_demod.grc` hierarchical block, GNU Radio screenshots of the demodulation results for VCM Mode 18 without and with a CMA equalizer are shown in Figures 12 and 13, respectively. Here, for Figure 13, a CMA equalizer of length 173 was used. As can be seen, the CMA equalizer dramatically improved the demodulation performance results in this case. The reason for this is that the input PSD from the data captures seen in the left-hand sides of Figures 12 and 13 suggest the presence of multipath [6]. Specifically, it appears as though the received sequence consists of the desired sequence (the main RRC pulse near the center frequency) plus a frequency-shifted and attenuated copy (the RRC pulse to the right of the main one). This effective multipath is likely the result of one of the images of the received sequence spectrum aliasing in-band to the main image [7].

GNU Radio screenshots of the demodulation results from VCM Modes 22 and 26 are shown in Figures 14 and 15, respectively. For each of these, a CMA equalizer using 173 taps was employed. From these figures, it can be seen that while the output constellations are more noisy than those for which the received data was internally generated or obtained via USRPs, the PLD constellations can still be made out. As with all other I/Q constellation plots shown here, there is a clustering around the BPSK constellation points from Figure 2 corresponding to the HDR symbols.

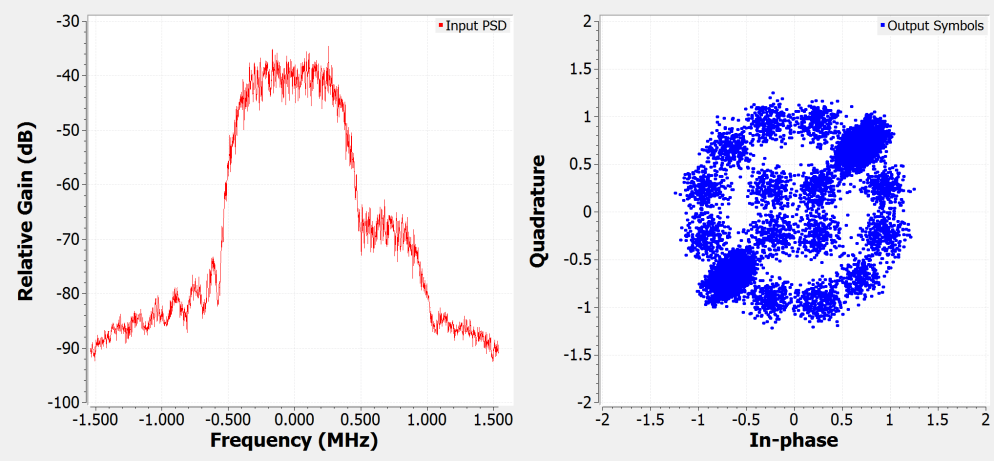

Figure 12. GNU Radio screenshot of VCM Mode 18 demodulation example using STRS generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols. No CMA equalizer was used here.

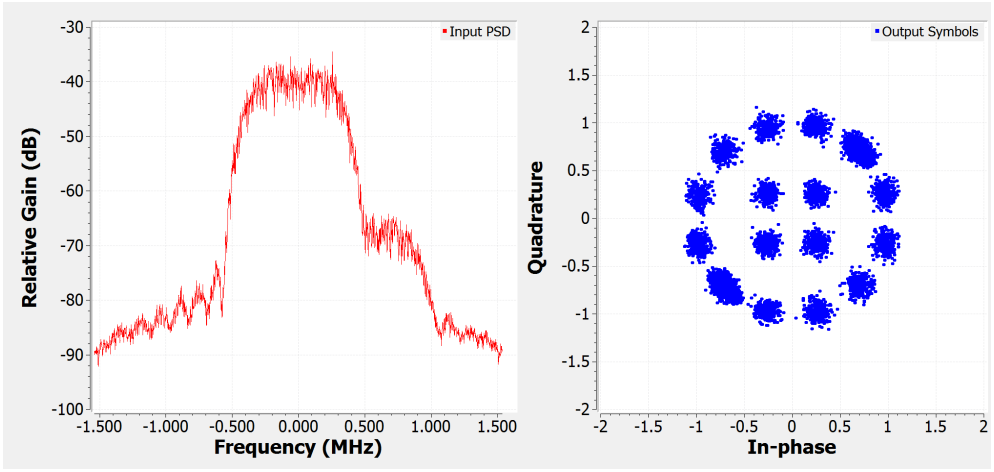

Figure 13. GNU Radio screenshot of VCM Mode 18 demodulation example using STRS generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

V. Conclusion

A receiver architecture for demodulating VCM PL frames was proposed. This topology, which exploited the presence of the known HDR FM sections, was shown to be amenable to the higher-order constellations, namely as 16/32/64-APSK, to be used for VCM/ACM. Demodulation results from data derived both in software and hardware showed the merits of the proposed architecture. In particular, for data obtained from the STRS, it was shown that the incorporation of a CMA equalizer between the symbol and carrier synchronization loops could greatly improve the demodulation performance.

Potential future extensions of the cognitive radio receiver include stripping off and decoding the FD bits, in order to identify the particular VCM mode being used for a specific VCM PL frame under observation. With such an extension in place, a next

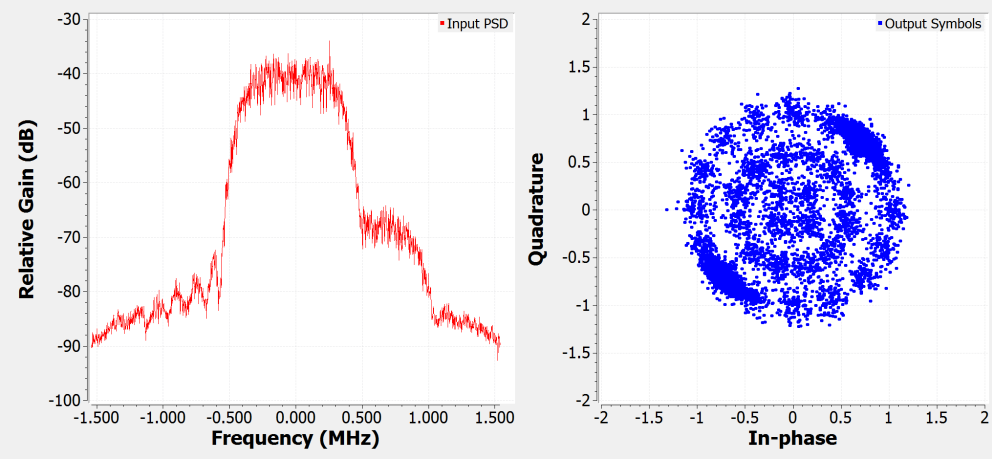

Figure 14. GNU Radio screenshot of VCM Mode 22 demodulation example using STRS generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

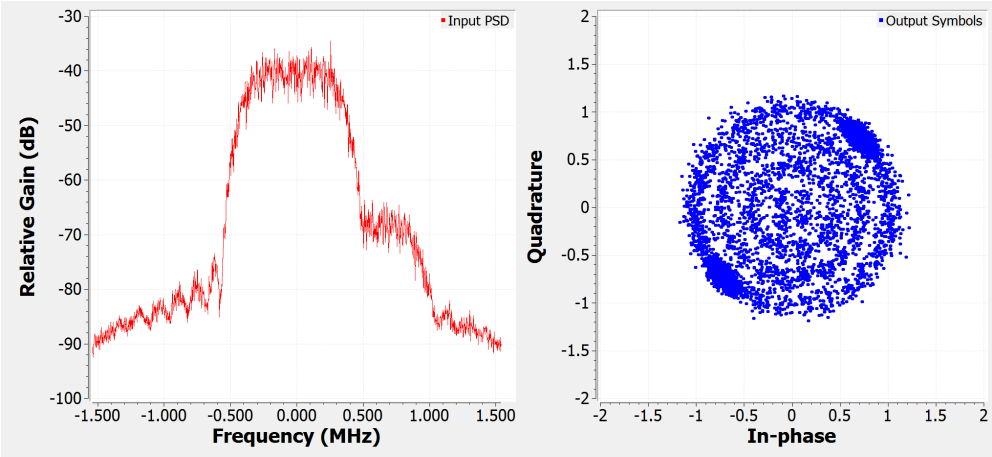

Figure 15. GNU Radio screenshot of VCM Mode 26 demodulation example using STRS generated data; left-hand side: PSD of input samples, right-hand side: I/Q constellation plot of output symbols.

logical step would be to incorporate decoders for the PLD data codewords. Once such decoders have been implemented, characterization of the bit error rate (BER) performance of the overall receiver architecture should be carried out for all of the data capture scenarios considered here.

Refinements to the STRS FPGA hardware will potentially be made to improve the fidelity of the transmitted VCM waveforms. In particular, ongoing efforts are being made to correct issues with the VCM modes employing the C2 error correcting code. Data captured from the STRS for these modes were shown to demodulate erroneously, and this problem was traced back to issues in the FPGA implementation used to generate the transmitted waveforms. Furthermore, for the modes that were shown to demodulate properly, in-band aliasing from attenuated images could be seen in the input power spectra. While the CMA equalizer in the receiver mitigated this issue, it is believed that a proper alteration to the FPGA implementation will eliminate it virtually entirely. This, in turn, will lead to improved demodulation performance overall.

References

[1] *TM Synchronization and Channel Coding*, Blue Book, Consultative Committee for Space Data Systems (CCSDS) Recommendation for Space Data System Standards CCSDS 131.0-B-3, Sep. 2017. [Online]. Available: https://public.ccsds.org/Pubs/131x0b3e1.pdf

[2] *Flexible Advanced Coding and Modulation Scheme for High Rate Telemetry Applications*, Blue Book, Consultative Committee for Space Data Systems (CCSDS) Recommendation for Space Data System Standards CCSDS 131.2-B-1, Mar. 2012. [Online]. Available: https://public.ccsds.org/Pubs/131x2b1e1.pdf

[3] *CCSDS Space Link Protocols over ETSI DVB-S2 Standard*, Blue Book, Consultative Committee for Space Data Systems (CCSDS) Recommendation for Space Data System Standards CCSDS 131.3-B-1, Mar. 2013. [Online]. Available: https://public.ccsds.org/Pubs/131x3b1.pdf

[4] *Low Density Parity Check Codes for Use in Near-Earth and Deep Space*, Silver Book, Consultative Committee for Space Data Systems (CCSDS) Experimental Specification CCSDS 131.1-O-2, Sep. 2007. [Online]. Available: https://public.ccsds.org/Pubs/131x1o2e2s.pdf

[5] H. Xie, S. Dolinar, M. Chase, S. Clancy, M. Kilzer, L. Torgerson, and A. Tkacenko, "Flight Test and Validation of Variable Coded Modulation Using SCaN Testbed," *Interplanetary Network Progress Report*, vol. 42-212, pp. 1–24, Feb. 2018. [Online]. Available: https://ipnpr.jpl.nasa.gov/progress_report/42-212/42-212B.pdf

[6] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. New York, NY: McGraw-Hill Education, Nov. 2007.

[7] F. J. Harris, *Multirate Signal Processing for Communication Systems*. Upper Saddle River, NJ: Prentice Hall PTR, May 2004.

[8] D. N. Godard, "Self-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Communication Systems," *IEEE Transactions on Communications*, vol. COM-28, no. 11, pp. 1867–1875, Nov. 1980.

Acronyms

M-ary modulation format where one of M symbols is transmitted at a time

16-APSK 16-ary amplitude and phase-shift keying

32-APSK 32-ary amplitude and phase-shift keying

64-APSK 64-ary amplitude and phase-shift keying

8-PSK 8-ary phase-shift keying

ACM adaptive coded modulation

APSK amplitude and phase-shift keying

AR4JA accumulate-repeat-4-jagged-accumulate

AWGN additive white Gaussian noise

BER bit error rate

BPSK binary phase-shift keying

CCSDS Consultative Committee for Space Data Systems

CFO carrier frequency offset

CMA constant modulus algorithm

CPO carrier phase offset

DEMUX demultiplexer

DTTL data transition tracking loop

DVB-S2 Digital Video Broadcasting - Satellite - Second Generation

ETSI European Telecommunications Standards Institute

FD frame descriptor

FM frame marker

FPGA field-programmable gate array

HDR header

I in-phase

I/Q in-phase / quadrature

ISS International Space Station

JPL Jet Propulsion Laboratory

LDPC low-density parity-check

MUX multiplexer

NCFO normalized carrier frequency offset

PL physical layer

PLD payload

PRBS pseudorandom binary sequence

PSD power spectral density

Q quadrature

QAM quadarature amplitude modulation

QPSK quadrature phase-shift keying

RRC root-raised-cosine

SCCC serial concatenated convolutional codes

SDR software-defined radio

STO symbol timing offset

STRS Space Telecommunications Radio System

USRP Universal Software Radio Peripheral

VCM variable coded modulation

XOR exclusive-or